Министерство просвещения Российской федерации

ФГБОУ ВО «Уральский государственный педагогический университет»

Институт математики, физики, информатики и технологий

Кафедра физики, технологии и методики обучения физике и технологии

Игровое приложение в жанре 2D-платформера под операционную систему Android

Выпускная квалификационная работа

Квалификационная работа
допущена к защите
Зав. кафедрой

_____ _____
дата подпись

Руководитель ОПОП

_____
подпись

Исполнитель:
Стрелкова Ксения Алексеевна
обучающаяся ПИ-1601-z группы

_____
подпись

Руководитель:
Алексеевский Петр Иванович
Старший преподаватель
Кафедры ИИТиМОИ

_____
подпись

Екатеринбург 2021

СОДЕРЖАНИЕ

СОДЕРЖАНИЕ 2
ВВЕДЕНИЕ 3
ГЛАВА 1. АНАЛИЗ ИНФОРМАЦИИ И ПОСТАНОВКА ЗАДАЧИ 5
1.1. Анализ возможностей использования мобильных устройств как игровых платформ 5
1.2. Анализ операционной системы Android 11
1.3. Формализованное описание технического задания 36
1.4. Назначение продукта разработки 38
2. РАЗРАБОТКА ИГРОВОГО ПРИЛОЖЕНИЯ 39
2.1. Описание процесса разработки игрового приложения 39
2.2. Описание структуры обучающего приложения 43
ЗАКЛЮЧЕНИЕ 47
СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ 48
ПРИЛОЖЕНИЕ 1 50
ПРИЛОЖЕНИЕ 2 53
ПРИЛОЖЕНИЕ 3 54
ПРИЛОЖЕНИЕ 4 56

ВВЕДЕНИЕ

В современном мире у большинства жителей есть смартфон на одной из ведущих конкурентную борьбу операционных систем iOS и Android. Android - открытая операционная система для мобильных телефонов, смартфонов, коммуникаторов, планшетных компьютеров, электронных книг, цифровых проигрывателей, наручных часов, нетбуков и смартбуков, основанная на ядре Linux и поддерживающая различные аппаратные платформы, такие как ARM, MIPS, POWER, x86.

И так как Android является открытой операционной системой, то представляет возможность множеству пользователей производить необходимый контент, то есть приложения для всех устройств. Разработка приложений под iOS ограничивает, так как программирование ведется в основном на устройствах компании Apple под управлением их операционной системы. Языков, которые используются в программировании под iOS на данный момент ограниченное количество, и поэтому разработку приходится начинать почти с нуля.

Совсем иначе обстоит ситуация с Android, где для разработки не требуется компьютер под управлением определенной операционной системы, где есть выбор между языками на котором писать программу, а порой можно даже не прибегать к изучению исходного кода и просто воспользоваться специализированными сервисами и потратив небольшое количество времени получить приложение.

Актуальность выпускной квалификационной работы состоит в том, что смартфон являются частью нашей повседневной жизни, они постоянно находятся рядом в наших руках и находясь в пути мы чаще всего используем игровые приложения для того чтобы скоротать время, что приводит нас к выводу о том, что разработка мобильных игр, для того, кто их разрабатывает является прибыльным делом и помогает реализовать свои мысли и идеи в ту среду, которая является частью жизни.

Предмет разработки - игровое приложение в жанре 2D-платформера под операционную систему Android.

Цель работы - создать игровое приложение в жанре 2D-платформера под операционную систему Android.

Для достижения поставленной цели в ходе работы необходимо решить следующие задачи:

- Проанализировать возможности использования мобильных устройств как игровых платформ;
- Проанализировать предметную область - операционную систему Android, её архитектуру, компоненты, структуру пакетов Java;
- Проанализировать существующие среды и языки программирования для операционной системы Android;
- Разработать игровое приложение в программной среде Construct 2;
- Создать обучающий ресурс, за основу которого использован продукт - игровое приложение в жанре 2D-платформера под операционную систему Android.

ГЛАВА 1. АНАЛИЗ ИНФОРМАЦИИ И ПОСТАНОВКА ЗАДАЧИ

1.1. Анализ возможностей использования мобильных устройств как игровых платформ

Мобильная игра - игровое приложение, предназначенное для мобильных устройств таких как сотовый телефон, смартфон, планшетный компьютер, портативная игровая консоль.

Впервые мобильная игра фигурировала в одном из первых сотовых телефонов, когда компания Siemens в 1994 году добавила на модель телефона S1 MARATHON возможность запустить игру Тетрис с помощью специального кода.

Однако первая мобильная игра, основанная на машинном коде - встроенная в состав прошивки устройства без возможности заменить или удалить, появилась лишь в 1997 году. Такой игрой стала Snake, в простонародье - Змейка, в ней она поглощала черные точки и тем самым становилась длиннее. Nokia 6110 стала первым телефоном, в который была встроена данная игра и она сразу же обрела множество фанатов, а компания продолжила встраивать данную игру на почти все свои сотовые телефоны.

С 1999 года начинается новая эра мобильных игр, так как вместе с выходом нового телефона Nokia 3310, компания начинает встраивать в свои телефоны уже четыре игры, среди которых была и обновленная версия Змейки.

В 2001 году Siemens выпускает свою 45-ю серию сотовых телефонов с совершенно новыми играми, среди которых Baloon Shooter, файтинг Battle Male, головоломка Super Mind, Moove the Box, Race Ace (одна из первых мобильных аркадных гонок), а также, любимейшая игра пользователей Siemens, геймплей которой чем-то напоминает Тетрис — Stack Attack [2.2].

В 2001 году появляется также и Java 2 Micro Edition, подмножество платформы Java для устройств, которые работали в режиме ограниченности ресурсов. Это позволило выйти мобильным играм на новый уровень и первый телефон, который поддерживал Java2ME стал Siemens SL45i.

Пользователи теперь могли самостоятельно решать какие игры они могут установить на свои устройства или удалить с них. Загрузить такие игры можно было с помощью интернета или же из других источников. Теперь к разработке игр для сотовых телефонов присоединились крупные игровые студии, такие как EA, Ubisoft, Konami, Disney, которые выпустили мобильные версии своих компьютерных хитов, таких как серия Assassin's Creed, The Sims, Need For Speed, PES, Command & Conquer и множество других популярных игр.

Вскоре к большим студиям присоединяются и другие, благодаря которым многие впервые увидели такие игры как Asphalt, Real Football, God Of War.

В 2007 году компания Apple выпускает свою мобильную операционную систему iOS и мобильный телефон теперь приобретает очертания смартфона. Google не отстает он своего уже теперь прямого конкурента и выпускает свою систему Android в 2008 году, которую могли установить на свои смартфоны многие компании. С этого момента начинается современная эра правления под управлением этих двух операционных систем, которые вытесняют с рынка устройства под другими операционными системами и конкурента у них пока еще не появилось.

Первой игрой, которая приобрела популярность на одновременно двух платформах была - Angry Birds, уже спустя всего год после релиза, она была скачена более миллиона раз. К 2010 году появляется игра Hungry Shark, которая была разработана на движке Unreal Engine 3, графика была одной из самых лучших, а сама игра завоевала множество наград. В это же время под смартфоны была адаптирован движок Unity. В след за игрой Hungry Shark появляются такие игры как Fruit Ninja, Galaxy on Fire 2.

В 2011 году на мобильные платформы вышел шутер N.O.V.A. 2 — Near Orbit Vanguard Alliance, игра установила новую планку графики и геймплея. Но длилось это недолго. Через 2 года были представлены гонки Real Racing 3

и Asphalt 8, графика которых до сегодняшних дней считаются одними из самых лучших среди аналогов [2.2].

Однако графика была не единственным, что выходит на новый уровень. В 2014 году выходит игра, которая вполне может конкурировать с игрой для персональных компьютеров Dota 2 - Vainglory. Игра в жанре MOBA - многопользовательская онлайновая боевая арена. И уже через год появляется игра в жанре RPG Aralon: Forge and Flame, которая поражает всех своим масштабом и многие признают в ней мобильную версию Skyrim.

За последние годы на мобильные платформы успели портировать такие, как Half-Life, Half-Life 2, Counter-Strike 1.6, Deus Ex, GTA SA, Bully, XCOM и другие не менее известные игры. Сейчас в магазинах приложений Android и iOS есть любые игры на все жанры. Мобильная индустрия игр развивается быстрыми темпами представляя новые разработки каждый день [2.2].

Операционная система (ОС) является главной особенностью смартфонов, которые отличают её от обычных сотовых телефонов. Теперь определяющим фактором при выборе постоянного устройства для повседневного назначения будет являться операционная система.

Самые распространенные операционные системы для смартфонов и платформ являются:

1) Symbian OS - до конца 2010 года, это была самая распространенная среди смартфонов операционная система, занимавшая большую часть рынка. К 2010 году на базе данной ОС остается лишь одна платформа, используемая в основном в устройствах компании Nokia - Series 60, а также в компании Samsung, которая использовала её в некоторых своих моделях;

2) BlackBerry OS - система, преимущественно распространённая в устройствах Соединенных Штатов. Причиной является то, что в смартфонах реализовано шифрование с использованием алгоритма AES, в котором не заинтересованы спецслужбы множества стран;

3) Windows Mobile OS - операционная система, выпущенная компанией Microsoft в 1996 году и которая до 2010 года, занимала довольно большой

сегмент мобильного рынка, но в настоящий момент компания приостанавливает поддержку данной ОС и отказывается от её дальнейшего развития;

4) Windows Phone 7 OS - пришла на смену Windows Mobile в 2010 году и прекратила своё развитие в 2019, когда компания Microsoft решает, что прекратит поддержку и просит своих клиентов о переходе на другие ОС. Использовала совершенно новый интерфейс, существенно отличающийся от Windows Mobile под названием Metro (или Modern UI), принципы которой были ранее использованы в дизайне интерфейса Windows Media Center, Zune и Xbox;

5) Windows Phone 8 OS - была вторым поколением операционной системы от компании Microsoft. Выход произошел в 2012 году с обновленным интерфейсом Metro. Данная операционная система использует новую архитектуру Windows NT, которая используется в операционных системах компании Microsoft предназначенной для персональных компьютеров. С операционной системы Windows Phone 7 было невозможно обновиться до Windows Phone 8, что соответственно запрещает возможность использовать новые приложения разработанные для второго поколения на предыдущем.

6) Palm OS - разработанная в 1996 году для КПК, была одной из популярных платформ, преимуществом которой была система распознавания рукописного ввода Graffiti. Мобильные телефоны на базе Palm OS в настоящее время мало распространены, но в своё время она была популярна, а последний телефон выпущенный с данной ОС был в 2007 году;

7) Linux OS - смартфоны на базе данной системы распространены в основном в Азии и широкого распространения среди мобильных устройств не получила, однако развитие Linux считается перспективным направлением;

8) Bada OS - разработана компанией Samsung в 2010 году, телефоны на платформе bada получили название «бадафоны» (bada phones) и позиционировались производителем как смартфоны. В 2012 году была объединена с Tizen — другой мобильной платформой, разрабатываемой

совместно с компаниями Intel, Asus и Acer. Выход был запланирован на 2013 год, предполагалось что после объединения будет довольно приличный каталог приложений, но уже в 2015 компания объявляет о закрытии проекта и прекращении поддержки разработчиков;

9) Android OS - операционная система для смартфонов, планшетных компьютеров, телевизоров, электронных книг, цифровых плееров, часов, автомобилей и нетбуков на базе ядра Linux и собственной реализации виртуальной машины Java, которая была разработана компанией Android Inc., и которую в дальнейшем купила компания Google. В 2007 году Google объявляет о создании Open Handset Alliance (OHA), которая и сейчас и тогда занимается поддержкой и дальнейшим развитием платформы. Кодовое имя каждой версии операционной системы Android представляло собой название какого-либо десерта, что прекратилось с выходом 10 версии. Android позволяет создавать приложения на основе Java, которые управляют устройством с помощью, разработанной Google библиотеки. Android Native Development Kit позволяет системе использовать библиотеки и компоненты приложений, написанных на C+ и других языках;

10) iOS (iPhone OS до 2010 года) - операционная система, разработанная компанией Apple для своих устройств, которая была выпущена в 2007 году. Изначально использовалась в устройствах iPhone и iPod Touch, а позже и в iPad и Apple TV. Доступна данная ОС исключительно для устройств, производимых компанией Apple. Интерфейс ОС основан на концепции прямого взаимодействия пользователя с использованием жестов «мультитач». Элементы управления состоят из ползунков, переключателей и кнопок.

На сегодняшний момент iOS значительно сдает в позиции популярной в мире операционной системы, так как iOS популярнее Android только в 8 регионах мира, так же из-за дороговизны устройств компании Apple есть страны, которые и вовсе не используют эту операционную систему, потому что не могут себе её позволить.

Android занимает лидирующую позицию на рынке и продолжает развиваться по всем фронтам, что привлекает внимание многих разработчиков создавать мобильные игры и приложения для данной ОС. Так же одним из плюсов разработки для Android является то, что разработка доступна каждому пользователю, даже если знаний в программном коде у человека минимальны, что позволяет продавать свои приложения и игры - каждому.

Платформа Android реализует идею адаптации программ общего назначения к мобильным устройствам, она многосторонняя и представляющая собой программный стек операционной системы на основе Linux, предназначенный для управления устройствами, памятью и процессами. В библиотеках Android содержатся функции, связанные с телефонией, видео, графикой, программированием пользовательских интерфейсов и некоторыми другими возможностями мобильного устройства [1.6].

HTC Dream - первый в мире смартфон, работающий на базе операционной системы Android с установленной версией 1.0, а после получивший обновления до версии 1.6 Donut. Смартфон имел форм-фактор горизонтального слайдера, а экран был сенсорным.

Ряд основных преимуществ, которые отличают операционную систему компании Google от аналогичных устройств на базе других платформ:

1) Открытость системы, так как устройство и его файловая система открыты пользователю, это дает возможность использовать устройство без ограничений - хранить разные типы файлов, скачивать их из сети интернет и делиться ими, что недоступно владельцам iOS.

2) Возможность устанавливать приложения из любых источников, а также возможность установки менять системные параметры с помощью прав суперпользователя

3) Возможность менять интерфейс, установив сторонний лаунчер, менять стандартные шрифты, кастомизировать устройство под свои собственные нужды.

4) Возможность активности приложений, запущенных в фоновом режиме до тех пор, пока это позволяет оперативная память.

5) Возможность подключения любых сторонних устройств ввода - проводной клавиатуры и мышки, жесткого диска.

6) Возможность устанавливать даже удаленные из официального магазина приложения, что недоступно пользователям компании Apple.

Однако кроме преимуществ у данной операционной системы присутствует и существенное ограничение - устаревание устройств. Компании, которые используют данную платформу в своих смартфонах постоянно стремятся выпускать все новые и новые устройства, совершенно не заботясь о том, чтобы обновлять их программное обеспечение - потому что это не выгодно. Часть устройств получает обновления и такие смартфоны по цене равняются с устройствами, выпускаемыми компанией Apple.

Также многие эксперты отмечают, что основанная на Java платформа не использует полный ряд преимуществ операционной системы Linux, таких как возможность появления большого числа приложений с полной версии на мобильную платформу из-за отсутствия общих серверов и библиотек. Помимо Java, компания Google предлагает использовать для разработки языки C/C++, но прибегать к ним советует в крайне редких случаях, к примеру, когда необходимо повысить производительность, использовать стороннюю библиотеку или программировать на низком уровне.

В октябре 2008 года был опубликован исходный код платформы Android, объем которого составляет около 2.1 гб, который лицензирован Apache 2.0, она позволяет любому использовать, модифицировать распространять код и по сей день.

1.2. Анализ операционной системы Android

Android - платформа, которая с точки зрения программиста, позволяет создавать код с помощью объектно-ориентированный язык программирования общего назначения Java.

Платформа операционной системы Android представляет собой программный стек для мобильных устройств, который включает операционную систему, программное обеспечение промежуточного слоя (middleware), а также основные пользовательские приложения, входящие в состав мобильного телефона, календарь, карты, браузер, базы данных контактов, сообщений SMS и прочее [1.4].

Архитектуру операционной системы принято делить на ряд уровней:

1) уровень ядра;

2) уровень библиотек и среды выполнения;

3)уровень каркаса приложений;

4) уровень приложений.

Уровень ядра представляет собой слой между основной частью программного стека и оборудованием. «Android основан на ядре Linux версии 2.6, но сама система Android не является Linux-системой в чистом виде. Система Android имеет некоторые отличия и содержит дополнительные расширения ядра Linux, специфичные для Android, — свои механизмы распределения памяти, взаимодействие между процессами и прочее [1.4].

Так как Android является гибкой, программный стек системы разработан для работы с дополнительными компонентами в мобильных устройствах. Данные компоненты полагаются на определенные аппаратные средства в устройстве.

Также на уровне ядра расположен набор драйверов для обеспечения работы с оборудованием мобильного устройства. Этот набор может отличаться в зависимости от производителя и модели устройства. Поскольку новое вспомогательное оборудование для мобильных устройств постоянно появляется на рынке, драйверы для них должны быть написаны на уровне ядра Linux для обеспечения поддержки оборудования [1.4].

Преимуществом использования ядра Linux как основы Android состоит в том, что ядро системы позволяет верхним уровням программного стека оставаться неизменными, несмотря на различия в используемом

оборудовании. Однако, хорошая практика программирования требует, чтобы пользовательские приложения корректно завершали свою работу в случае вызова ресурса, являющегося недоступным, к примеру, встроенная видеокамера или сенсор, не присутствующего в определенной модели устройства [1.4].

Следующим уровнем над ядром является уровень библиотек, включает библиотеки C/C++, использует различные компоненты операционной системы. Библиотеки бывают системные и функциональные.

Системная библиотека базируется на BSD (Berkeley Software Distribution). Компания Google разработала собственную версию системной библиотеки libc - Bionic специально для мобильных устройств на основе Linux, это было необходимо для обеспечения быстрой загрузки библиотеки в каждый процесс и, следовательно, библиотека должна была иметь маленький размер. Данная библиотека Bionic имеет размер около 200 Кбайт, что в два раза меньше размера стандартной библиотеки Linux glibc. Кроме того, необходимо было учитывать ограниченную мощность центрального процессора мобильного устройства - это означает, что библиотека должна быть оптимизирована для максимального быстродействия. В данный момент это уже не актуально, современные мобильные устройства практически сравнялись по мощности процессора с нетбуками, но еще несколько лет назад это являлось серьезной проблемой [1.4].

Библиотека Bionic имеет встроенную поддержку важных для Android системных служб и регистрацию системных событий, но в то же время она не поддерживает некоторую функциональность, например, исключения C++, и несовместима со стандартом POSIX и GNU libc [1.4].

Уровень каркаса приложений находится выше системных библиотек и располагает основными службами для управления жизненным циклом приложения, пакетами и ресурсами.

К API, которые используются основными приложениями - программист имеет полный доступ. Архитектура этих приложений разработана с целью

упрощения многократного использования компонентов. Любое разрабатываемое приложение может использовать возможности базовых приложений и, соответственно, любое другое стороннее приложение может использовать возможности вашего приложения (с учетом установленных разрешений), этот же самый механизм позволяет многократно использовать уже разработанные компоненты [1.4].

Уровень приложений представляет собой устройство, которое поставляется с определенными набором приложений - почтовый клиент, смс-клиент, календарь, навигационные карты, браузер, контакты и прочее. При разработке приложений программисты имеют полный доступ ко всей функциональности операционной системы. Архитектура приложений построена так, чтобы было легко использовать основные компоненты, предоставляемые системой, также есть возможность создавать свои компоненты и предоставлять их в открытое использование [1.4]

Android приложение состоит из компонентов, которыми система распоряжается как ей необходимо, запуская и управляя ими. Компоненты делятся на четыре типа: activity, service, broadcast receiver, content provider.

Activity или активность, которая запускается первой, считается основной.

Activity - это компонент, который представляет собой визуальный пользовательский интерфейс для приложения или окно. Окно как правило полностью заполняет экран мобильного устройства, но может иметь размеры меньше, чем у экрана. Activity также может использовать дополнительные окна, к примеру - всплывающее диалоговое окно, которое запрашивает пользовательский ответ для основного Activity, или окно уведомления о каком-либо событии в приложении или системе. Все Activity реализуются как подкласс базового класса Activity. Приложение может содержать несколько Activity, и каждый будет независим от других. Открывая новое Activity работа предыдущего приостанавливается, а оно само вносится и сохраняется в стек [1.4].

Служба или Service работает как фоновый процесс, выполняет сетевые запросы к веб-серверу и обрабатывает информацию, запуская уведомления и прочее.

Service - это компонент, который не имеет визуального интерфейса пользователя и выполняется в фоновом режиме в течение неопределенного периода времени, пока не завершит свою работу. Компонент аналогичен службам в настольных операционных системах и приложения могут подключаться к компоненту Service или запускать его, если он не запущен, а также останавливать уже запущенные компоненты. Подключившись к Service, можно обратиться к функциям этого компонента через интерфейс предоставляемый этим компонентом [1.4].

Компонент Broadcast Receiver или широковещательные сообщения делают приложение более открытым, передавая события и используя сообщения.

Broadcast Receiver - компонент для получения внешних событий и реакции на них. Инициализировать передачи могут Service и другие приложения. Приложения могут иметь несколько компонентов Broadcast Receiver, чтобы ответить на любые объявления, которые оно считает важными. Компонент не имеет пользовательского интерфейса, однако может запустить Activity или службу, выдать в ответ на информацию, которую они получают, или показать уведомление на экране мобильного устройства, чтобы предупредить пользователя о наступившем событии [1.4].

Компонент Content Provider или поставщик содержимого представляет собой оболочку содержащую данные.

Content Provider делает определенный набор данных, используемых приложением, доступным для других приложений. Данный компонент является своеобразным посредником между хранилищем данных и клиентским приложением. Данные в Android могут быть сохранены различными способами: в файловой системе, в базе данных SQLite или любым другим способом. Компонент для безопасного доступа к данным использует

механизм разрешений, это означает, что можно сконфигурировать собственный Content Provider, чтобы разрешить доступ к данным из других приложений, а также использовать Content Provider другого приложения для обращения к его хранилищу данных [1.4].

Компонент Intent или намерение - механизм, описывающий одну операцию - выбор фотографии, отправление письма, совершение звонка, запуск браузера, переход на указанный адрес.

Особенность платформы Android состоит в том, что одно приложение может использовать элементы других приложений при условии, что эти приложения разрешают использовать свои компоненты, при этом ваше приложение не включает код другого приложения или ссылки на него, а просто запускает нужный элемент другого приложения. Поэтому, в отличие от приложений в большинстве других систем, у приложений Android нет единственной точки входа для запуска всего приложения, аналогичной, например, функции main() в C-подобных языках программирования и для реализации такого использования компонентов других приложений, система должна быть в состоянии запустить процесс для приложения, в котором находится требуемый компонент, и инициализировать нужные ей объекты [1.4].

Пользовательский интерфейс Android используется в рамках операционной системы и при программировании используется объявление интерфейса в файлах XML, где представление загружаются в приложение с пользовательским интерфейсом.

Экраны или окна часто называют деятельностью, которая включает в себя несколько типов, в которой пользователи представляют собой элемент процесса. Виды или представления являются основным элементом, которые находятся в пользовательском интерфейсе. Формы объединены в группы видов, для внутренней организации которого используется течение длительного времени концепция холста и взаимодействие с системой пользователем.

Ключевым понятием является управление жизненным циклом окна событий (окон деятельности). Система Android может управлять ситуацией пока пользователь выполняет определенные действия чтобы скрыть, восстановить, остановить или закрыть окно.

Android SDK - это универсальная среда разработки приложений, отличительной чертой которой является наличие широких функциональных возможностей, позволяющих запуск тестирования и отладки исходного кода в режиме совместимости с различными версиями операционной системы.

Элементы управления в операционной системе Android, которые наиболее используются в Android SDK.

1) текстовые;

2) кнопки;

3) списки;

4) таблицы;

5) дата и время;

6) карта;

7) галерея;

8) счетчик.

Текстовые элементы являются самыми простыми из элементов управления, они предназначены для отображения текста, возможности редактирования текста

TextView элемент, который предназначен для отображения текста, он не позволяет редактировать [1.6]. Является простым и удобным, в нем может содержаться гиперссылка, для которой можно установить значение web для свойства autolink, в таком случае элемент найдет и выделит URL.

Edittext элемент управления является субклассом TextView. Как понятно из названия, элемент управления EditText позволяет редактировать текст [1.6]. Не многофункционален, но ему можно указывать свойства, которые позволят исправлять ошибки, устанавливать заглавные буквы в

начале предложений, установить возможность создания поля для ввода пароля.

AutoCompleteTextView элемент управления, вариант TextView, в котором предусмотрена функция автоматического завершения ввода [1.6]. В данном элементе предусмотрена функция автоматического завершения ввода, помогает пользователю в выборе языка.

Mu1tiAutoComp1eteTextView похож в использования на AutoComp1eteTextView. Разница заключается в том, что необходимо указать элементу управления, с какого места снова предлагать варианты ввода [1.6]. Данный элемент в отличие от предыдущего предлагает варианты только для законченного варианта текста, который может быть введен пользователем.

Элемент управления кнопка представляет собой набор типичных, распространённых и в других системах кнопок. Это класс Android, в котором хранится информация об обычных кнопках - это android.widget.Button. О данном типе кнопок можно сказать не так много, кроме того, что они используются для работы с событиями на нажатие (click cvents) [1.6]. В нем регистрируется событие, которое будет происходить в ответ на нажатие пользователем кнопки.

Изображения-кнопки в Android содержатся в android.widget.Image-Button, работа с ними практически не отличается от использования обычных кнопок [1.6]. Изображение, которое будет отображаться, можно установить динамически или с помощью изменения XML-файл шаблона.

ToggleButton элемент, который подобно флажку (checkbox) или переключателю (radiobutton) - кнопка, которая может оказываться в одном из двух состояний: активна (On) или неактивна (Off). По умолчанию на кнопке написано On (Включено), если она активна, и Off (Выключено) - если нет [1.6]. С помощью такой кнопки можно установить фоновый процесс, который можно запустить или остановить.

Checkbox или флажок - это элемент управления, который присутствует практически во всех наборах виджетов. Флажки поддерживаются и в HTML,

и в JFC, и в JSF. Checkbox - это элемент, который может переключать систему между двумя состояниями [1.6]. В таком элементе можно задействовать логику, при которой флажок будет включаться и выключаться, а затем нужно реализовать метод, который будет вызываться при изменении состояния флажка.

Radiobutton (переключатель) - это неотъемлемая часть любого инструментария, предназначенного для создания пользовательских интерфейсов, они используются в тех случаях, когда от пользователя требуется выбрать только один элемент из предлагаемого списка [1.6]. Такие элементы объединяют в группы, где в каждой группе будет отмечен только один элемент.

ListView позволяет отобразить вертикальный список элементов, обычно используется при написании нового явления, дополняющего android.app.ListActivity [1.6]. В данном элементе вызывается метод setlistAdapter ( ) для явления, например запрашиваем список контактов, а после создаем проекцию, при помощи которой можно выбрать имена контактов, в проекции задаются интересующие пользователя столбцы.

GridView - элемент управления, который отображает информацию в форме таблицы. Обычно GridView используется, чтобы задать сетку таблицы в XML-шаблоне, а затем связать данные с таблицей при помощи android.widget.ListAdapter [1.6]. Данную таблицу можно заполнить изображениями и прочим контентом.

Android также предлагает такие элементы управления, как DatePicker, TimePicker, AnalogClock и DigitalClock [1.6]. DatePicker используется для выбора даты, TimePicker для выбора времени, AnalogClock и DigitalClock представляют собой отображение часов, в них нельзя изменять дату и время.

MapView или com.google.android.maps.MapView - это элемент управления, который может отображать карту. Этот элемент управления можно инстанцировать либо через шаблон XML, либо через код, но то явление, которое будет использовать этот элемент, должно бьrrь добавлено к

MapActivity. Последний отвечает за обработку многопоточных запросов и выполняет такие задачи, как загрузка карты, осуществление кэширования и прочее [1.6].

Gallery - это элемент, который представляет собой горизонтальный прокручиваемый список, позволяющий поместить в фокус тот элемент списка, который находится в центре. Фотогалерея обычно элемент управления, который работает в сенсорном режиме [1.6].

Работа со счетчиками также похожа на работу со списками, то есть получаете ссылку на счетчик, а затем вызываете метод setAdapter ( ) для занесения данных, потом регистрируете события, которые должны происходить при выборе элемента [1.6].

В разработке приложений для Android используется высокоуровневый прикладной интерфейс программирования Java, при помощи которого можно создавать приложения для конечных пользователей.

Набор средств разработки (SDK) для Android поставляется с Android Studio, плагином, который называется набором инструментальных средств для разработки на Android (ADT), данный инструмент предназначен для разработки IDE (IDE) создания, отладки и тестирования приложений на Java.

Android SDK может использоваться без ADT. Вместо инструментов, в нем можно использовать инструменты командной строки. Эмулятор поддерживает использование обоих подходов, и с его помощью можно запустить, исправить и проверить приложения, 90% разработки приложений может быть завершена, даже без использования реального устройства. Полностью функциональный эмулятор для Android воспроизводит наиболее изученные характеристики устройства, среди которых есть те, которые не могут быть имитированы в эмуляторе - USB-подключение, работа камеры и видео, имитация работы наушников, батареи и технологии Bluetooth.

Android Emulator базируется на технологии с открытым исходным кодом «имитация» процессора под названием QEMU. Эта самая технология может эмулировать одну операционную систему на другой, независимо от того, какая

используется процессором, а QEMU обеспечивает эмуляцию уровня процессора.

Пакеты, входящие в состав Android SDK, позволяют составить представление о платформе Android. Поскольку Android отличается от стандартного дистрибутива SDK, важно знать, какие пакеты поддерживаются, а какие - нет. Краткое описание важных пакетов, которые входят в состав Android SDK:

1) Android.App – представляет собой реализацию модели приложения для Android, среди основных классов оно является приложением, которое описывает начальную и конечную семантику, а также целый ряд классов, связанных с явлениями, элементы управления, диалоговых окон, окон предупреждений и уведомлений;

2) Android.Bluetooth – содержит в себе классы для работы с технологией Bluetooth, основными такими классами являются - VluetoothAdapter, BluetoothDevise, BluetoothSocket, BluetoothServerSocket и BluetoothClass. BluetoothAdapter класс может быть использован для управления Bluetooth-адаптером, установленным на локальном компьютере. Такой адаптер позволяет включить, отключить, а также запустить процесс обнаружения. BluetoothDevise является классом с удаленным устройством Bluetooth, к которому можно подключиться. Bluetooth для связи имеет два сокета, которые используются между устройствами. Класс Bluetooth является типом устройства Bluetooth, к которому можно подключиться;

3) Android.content - представляет собой концепции, которые связанны с поставщиками контента. Поставщик позволяет суммировать обмен и хранение данных, а также этот пакет реализует основные идеи относительно намерений и равномерного идентификаторов ресурсов (URI) в Android;

4) Android.content.pm - предоставляет собой классы для работы, связанные с помощью диспетчера пакетов, он содержит информацию о разрешениях, установленных пакетов, установленных поставщиками, услуг и компонентов, таких как акции, а также установленных приложений;

5) Android.content.res - предоставляет собой доступ к файлам ресурсов, как структурированными, так и неструктурированным. AssetManager основные классы (для неструктурированных ресурсов) и материальных ресурсов;

6) Android.database - реализует собой идею абстрагирования базы данных, а основной интерфейс называется Cursor;

7) Android.database.sqlite - представляет собой концепцию пакета базы данных Android, с использованием в качестве физической базы данных SQLite. Основные такие классы - SQLiteCursor, SQLiteDatabase, SQLiteQuery, SQLiteQueryBuilder и SQLiteStatement. Однако, в основном приходится работать с классами абстрактного пакета базы данных Android;

8) Android.gesture - в данном пакете располагаются все классы, а также интерфейсы, которые необходимые для работы с определенными пользователем жестами. Основные такие классы Gesture - GestureLibrary, GestureOverlayView, GestureStore, GestureStroke, GesturePoint. Класс Gesture также является подборкой GestureStrokes и GesturePoints.

В библиотеке GestureLibrary собраны жесты, они хранятся в GestureStore. Имена этих жестов таковы, что система может идентифицировать их как действия:

1) Android.graphics, который содержит класс Canvas, Camera, Color, Matrix, Movie, Paint, Path, Rasterizer, Shader, SweepGradient nTypeFace;

2) Android.graphics.drawable, который предназначен для работы с протоколами рисования и фоновых изображений, обеспечивает эффекты анимации при работе с рисованными объектами;

3) Android.graphics.drawable.shapes, которые предназначены для работы с контурами, в том числе ArcShape, OvalShape, PathShape, RectShape и RoundRectShape;

4) Android.hardwa, который позволяет использовать так называемые естественные классы, предназначенные для работы с камерой. Класс камеры также является распространенным устройством - камерой, а класс

android.graphics.Camera - его графическая концепция, не имеющая никакого отношения к реальной физической камере;

5) Android.location, который содержит в себе классы Address, GeoCoder, Location, LocationManager и LocationProvider. Класс Address представляет собой упрощение замещённого Language XAL (расширяемый язык адреса). Geocoder, он позволяет узнать адрес координат объекта (широта и долгота), и наоборот, а в Location представлена информация о широте и долготе;

6) Android.media, который содержит в себе классы MediaPlayer, MediaRecorder, Ringtone, AudioManager и FaceDetector. Такой класс, как MediaPlayer предназначен для потоковой поддержки аудио и видео, а класс Ringtone используется для воспроизведения коротких аудиоотсчетов, которые могут быть использованы в мелодии или уведомлений. Класс AudioManager отвечает за регулировку громкости, а класс FaceDetector может быть использован для обнаружения человеческих лиц на точке (растровые) рисунки;

7) Android.net является реализацией базовой сети на уровне сокетов API. Основные её классы включают Uri, ConnectivityManager, локальный сокет и местный ServerSocket, также следует отметить, что Android поддерживает уровень HTTPS-браузера и сетевой уровень, а кроме того, Android поддерживает JavaScript в браузере;

8) Android.net.WiFi отвечает за управление подключением Wi-Fi. Основные его классы WifiManager, который отвечает за составление списка настроенных сетей и работает с текущей активной сетью Wi-Fi и WifiConfiguration;

9) Android.OpenGL содержит в себе вспомогательные классы, которые используются при выполнении операций OpenGL ES. Классы OpenGL ES являются частью другого набора пакетов, взятых из JSR 239.

10) Android.OS является службой операционной системы, доступ к которой осуществляется с помощью языка Java. Некоторые её важные классы - BatteryManager, Биндер, FileObserver, Хэндлер, Looper и PowerManager.

Binder класс, который обеспечивают обмен информацией между процессами. Класс FileObserver ведет учет изменений в файлах, а класс Handler используется для выполнения задач в потоке сообщений, и Looper с которого начинается само сообщение потока, его пакеты:

1) Android.preference, который позволяет приложениям предоставить пользователям возможность управлять своими настройками для этого приложения в единой форме. Основные его классы PreferenceActivity, PreferenceScreen;

2) Android.provider, который включает в себя набор готовых контент-провайдеров, связанных с android.content.ContentProvider. Поставщики контента - Контакты, MediaStore, браузер и настройки. Этот набор, содержит интерфейсы и классы, которые содержит метаданные для описания базовой структуры данных;

3) Android.sax, который обеспечивает эффективный набор простых API для XML (SAX), вспомогательные классы, предназначенные для синтаксического анализа. Основные её классы Element, RootElement некоторые ElementListener интерфейсы;

4) Android.speech, который содержит константы для распознавания речи. Входит только в версии 1.6 и выше;

5) Android.speech.tts, который обеспечивает поддержку для преобразования текста в речь. Основной её класс - TextToSpeech. Android содержит механизм PICO TTS (преобразования текста в речь, синтезатор речи) производства SVOX;

6) Android.tellphony, который содержит классы CellLocation, PhoneNumberUtils и TelephonyManager. Класс TelephonyManager используется для определения местоположения, из которого был сделан вызов, номер телефона, имя поставщика услуг, тип сети, тип телефона и серийный номер модуля идентификации абонента (Subscriber Identity Module, SIM);

7) Android.tellphony.GSM, который позволяет собирать информацию об адресах ячеек, основанных на местоположении данных вышек сотовой связи,

но также содержит классы, ответственные за работу с SMS-сообщениями. Данный класс определяет глобальную систему мобильной связи во имя этого пакета называется GSM, как оригинальные коротких стандартов обмена сообщениями (SMS) (Глобальная система мобильной связи);

8) Android.telephony.CDMA, который поддерживает CDMA-телефонию;

9) Android.text, который содержит классы для обработки текста;

10) Android.text.method, который предоставляет классы для ввода текста в различные элементы управления;

11) Android.text.style, который обеспечивает разнообразие методов обработки текста;

12) Android.Utils, который содержит классы, DebugUtils, TimeUtils и Xml;

13) Android.view, который содержит классы, меню View, ViewGroup, а также некоторые из процессов слушателей и обратных вызовов;

14) Android.view.animation, который обеспечивает поддержку анимации в структуре промежуточных кадров;

15) Android.view.InputMethod, который реализует ввод-вывод системной архитектуры. Содержится только в версиях 1.5 и выше;

16) Android.WebKit, который содержит классы, связанные с веб-браузером. Среди основных её классов - WebView, CacheManager и CookieManager;

17) Android.widget, который содержит все классы элементов управления пользовательского интерфейса, которые получены главным образом с точки зрения класса. Основные виджеты – это Кнопка, Галочка, хронометр, AnalogClock, DatePicker, EditText, ListView, FrameLayout, GridView, ImageButton, MediaController, ProgressBar, RadioButton, RadioGroup, RatingButton, скроллер, Scrollview, Spinner, TabWidget, TextView, TimePicker, VideoViewn ZoomButton;

18) com.google.android.maps содержит в себе класс MapView, MapController и MapActivity, необходимые для работы с Google Maps [2.10].

Вышеуказанные пакеты очень важны при работе с Android, исходя из этого списка, можно получить представление о глубинном строении платформы Android.

В целом, Android Java API включает в себя более 40 пакетов и более 700 классов, однако, все эти многочисленные пакеты составляют богатую вычислительную платформу, предназначенную для написания программ для мобильных устройств.

Для того чтобы разработать приложение для операционной системы Android в данный момент даже не требуются знания в области программирования и исходного кода, достаточно будет зайти на специализированный ресурс, выбрать один из готовых шаблонов и усовершенствовать его под свои задачи.

Поэтому первая среда программирования - это онлайн-конструкторы, сайты где можно бесплатно или за определенную денежную сумму в течение нескольких часов создать готовый продукт и разместить его в магазине приложений PlayMarket. Такая среда является интуитивно понятной, обладает логической и последовательной работой в редакторе, позволяет добавлять разнообразные модули (PUSH и Alert уведомления, календарь, онлайн-записи, объявления и другое). Примеры таких онлайн-конструкторов: AppsGeyser, TheAppBuilder, Appsmakerstore [2.3].

Вторая среда программирования AppMaker - сервиса Google, предназначенного для лёгкого создания приложений для различных нужд организаций. AppMaker позволяет посредством перетаскивания виджетов создавать приложения с интерфейсом, выполненным в вещественном дизайне. Затем эти приложения можно улучшать скриптами, а также контентом на базе HTML, CSS, JavaScript и JQuery. AppMaker обеспечивает пользователю low-code среду, то есть возможность создать приложение с помощью графического пользовательского интерфейса и выбора конфигурации, а не с помощью стандартных процедур программирования [2.3].

Третья среда программирования уже представляет собой работу с исходным кодом - Android Studio, которая является интегрированной средой разработки (IDE) для работы с платформой Android. Она поддерживает работу с несколькими языками программирования, к которым относятся самые популярные - C/C++, Java. Позволяет разрабатывать приложения не только для смартфонов/планшетов, но и других устройств под управлением Android. Позволяет тестировать приложение на предмет ошибок и обладает удобным редактором кода [2.1].

В подавляющем большинстве случаев код этих приложений пишется на языке программирования Java, который постоянно развивается, как и развиваются среды разработки мобильных приложений, использующие этот язык программирования в качестве основного.

К одной из наиболее используемых и развитых в функциональном плане сред разработки относится программный продукт Android Studio, особенности работы с которым будут рассмотрены далее.

Android Studio представляет собой интегрированную среду разработки мобильных приложений (первая стабильная версия которой вышла в 2014 г.) для операционной системы Android, где одним из языков программирования официально является язык Java.

Процесс установки и (или) настройки Android Studio достаточно прост и интуитивно понятен большинству пользователей. В системе реализован механизм оповещения о доступных обновлениях, которые устанавливаются автоматически.

Помимо Android Studio также может потребоваться установка и (или) настройка эмулятора Android-устройств, необходимого для тестирования разработанных приложений, например, Genymotion.

Разработка приложения в Android Studio формально состоит из двух этапов: создания оконных форм, или Activity, и кода программных модулей, что осуществляется на рабочей области, которая позволяет переключаться между ними в процессе выбора файлов, входящих в структуру приложения.

Создание нового мобильного приложения в Android Studio начинается с выбора пункта меню File - New - New Project.

Далее требуется задать имя проекта, а также путь к папке, где он будет расположен.

Затем нужно выбрать минимальную версию операционной системы Android, необходимую для запуска приложения.

После чего Android Studio попросит выбрать шаблон приложения из числа имеющихся. В целях обучения отлично подходит шаблон под именем Empty Activity, или пустое окно. Как только шаблон будет выбран следует задать имя главного окна и нажать кнопку «Finish».

В левой части окна Android Studio отображается структура приложения в виде папок с файлами, каждый из которых можно открыть для просмотра или редактирования в правой части окна. Переключение между открытыми файлами осуществляется путем выбора соответствующих им закладок.

В структуру приложения, разрабатываемого в Android Studio, входят следующие основные элементы:

1) файл AndroidManifest.xml, который находится в папке manifests и предназначен для редактирования глобальных настроек приложения, таких как: имя приложения, выбор главного окна и стиля оформления, и пр.;

2) файлы с расширением .java, расположенные в папке java. Данные файлы представляют собой программные модули, код которых пишется на языке программирования Java;

3) файлы с расширением .xml, расположенные в папке res - layout и содержащие настройки для окон и элементов управления, используемых в приложении, в формате XML (от англ. eXtensible Markup Language).

4) файл string.xml, находящийся в папке res - values, используется для хранения текстовых констант, используемых в приложении.

Проектирование оконной формы осуществляется путем размещения на ней различных элементов управления, или View, таких как радиокнопки,

текстовые поля, переключатели, кнопки и пр., расположенные на палитре инструментов, размещенной в левой части редактора оконных форм.

Чтобы добавить нужный элемент управления на форму, необходимо выбрать его в списке, щелкнув по нему мышкой, а затем перетащить его на форму.

После размещения элемента управления, или компонента, необходимо произвести настройку его свойств на соответствующей вкладке в правой части редактора оконных форм, кроме того можно с помощью мыши менять его положение и (или) размеры.

Чтобы изменить значение какого-либо свойства, требуется щелкнуть мышкой по полю, напротив его названия, а затем ввести нужное значение.

Все компоненты, размещенные на оконной форме, отображаются в дереве объектов, которое находится в левом нижнем углу. Переключение между режимами редактирования оконных форм: в формате XML или в режиме редактора производится путем переключения соответствующих закладок: Text и Design, размещенных в нижней части среды разработки Android Studio.

Сами по себе, элементы управления View, размещенные на форме, не выполняют каких-либо действий. Для этого необходимо запрограммировать их реакцию на конкретные события, например, на нажатие кнопки мышки. Так, в коде программы описываются методы обработки каких-либо событий, которые затем следует привязать к компоненту в свойстве onClick, путем указания их имени. Каждому элементу управления можно присвоить свое собственное имя, по которому к нему можно обращаться из программы, в свойстве ID. программный мобильный приложение код

Элементы управления, или View, по умолчанию размещаются на так называемых Layout (слоях или группах View), из которых состоит Activity. Каждый слой Layout обладает своими отличительными особенностями, связанными с размещением на нем View. По умолчанию в Android Studio версии 2.3 используется Constraint Layout. К основным элементам управления,

помещаемым на форму Activity, относятся: TextView (текстовая надпись), Button (кнопка), editText (поле для ввода текста), checkBox (поле для установки/снятия флажка) и прочее, которые будут рассмотрены во второй части статьи. Кроме того, Android Studio обеспечивает программистов большим количеством функциональных возможностей, облегчающим разработку, тестирование и отладку мобильных приложений.

Unity3D - среда разработки игр, которая является межплатформенной. Данная платформа позволяет создавать приложения, работающие на 25 платформах, основным преимуществом которой является визуальная среда разработки.

Unity3D можно назвать движком номер один для создания игр, как мобильных, так и десктопных или приставочных. Данный движок поддерживает все актуальные операционные системы. Благодаря использованию OpenGL с помощью Unity3D можно оформлять игру по последнему слову современного программирования. Работу с движком облегчает большое количество библиотек, содержащих скрипты практически на любой случай. Кодирование также возможно на языках C#, Javascript. В пакет встроены редакторы, позволяющие создавать собственные 3D модели, звуки, ландшафты, физику и многое другое, что наверняка потребуется в процессе разработки [2.8].

Unreal Engine 4 - игровой движок, разработанный компанией Epic Games, который изначально предназначался для разработки шутеров от первого лица, а позже успешно доработанный и применяемый в играх самых различных жанров.

Unreal Engine 4 может смело претендовать на вторую позицию в рейтинге. Все благодаря тому, что именно на нем написаны Mass Effect, Batman: Arkham City и Mortal Kombat. Как и Unity3D, позволяет создавать игры для любых актуальных операционных систем. Для визуального программирования используется уникальная рабочая среда Blueprint, а код можно писать на C++. Обладает очень богатым и гибким инструментарием,

включающим редакторы анимации, частиц, ландшафтов, визуальных эффектов [2.8].

GameMaker - написанный на Delphi конструктор игр, рассчитанный на создание двумерных игр любых жанров, разработанный компанией YoYo Games.

Является визуальным редактором игр, не требующим работы с кодом, однако не исключающий подобной возможности - для этого предусмотрен собственный язык, синтаксически напоминающий C++ и Pascal. Позволяет создавать простые 3D игры, хотя основной упор сделан всё-таки на 2D, также содержит обширную библиотеку звуков и изображений, имеет встроенный редактор уровней и событий. В число поддерживаемых платформ входят Windows, Android, Linux, Mac [2.8].

Construct 2 - представляет собой программу объединяющую в себе игровой движок и интегрированную среду разработки, разработанную компанией Scirra.

Движок позволяет создавать только 2D игры, но при этом не требует таких серьезных познаний в программировании, как Unity3D или Unreal Engine 4. Интерфейс программной среды интуитивно понятен, сам процесс разработки больше напоминает рисование: вы добавляете и редактируете объекты прямо на холсте будущей игры, после чего подключаете готовую анимацию и физику. Настройки гибкие, позволяют сконфигурировать среду под собственные запросы. Также доступна обширная библиотека плагинов, еще больше расширяющая возможности разработчика. Позволяет создавать игры для Android, iOS, Windows, Linux и Mac, а при необходимости - выполнять их кроссплатформенный экспорт [2.8].

Программа позволяет создавать 2D-игры различных жанров и сложности, а интерфейс программы представляет собой визуальный редактор, в котором имеются события и действия, создающие логику игры.

Редактор написан на языке C++, а игры кодируются в Javascript. Экспорт проекта JavaScript минифицируется то есть уменьшается размер исходного

кода путём удаления ненужных символов без изменения его функциональности, однако есть возможность подключить Javascript Plugin SDK и модифицировать код вручную. Программа представляет собой модульный дизайн, поэтому любые плагины или поведения, которые не используется в проекте, не включаются в скрипт, что помогает оптимизировать проект и сократить его вес [2.9].

Construct 2 первый игровой движок, который официально поддерживает эффекты частиц (particle effects) на HTML5.

Главная особенность программы поведения (behaviours) - это предварительные функциональные возможности. К примеру, добавив поведение Platform к игроку и поведение Solid к платформе, игрок сможет передвигаться и прыгать на платформе. Виды поведений, которые присутствует в Construct 2:

1) 8 Direction movement или Движение по 8 направлениям, оно позволяет объекту двигаться с помощью клавиш-стрелок или кнопок, управляемых с помощью Touch.

2) Bullet movement или Движение пули, поведение которое просто двигает объект прямо под готовым углом. Применяется для пуль игрока. Также хорошо подходит для движения монстра, поскольку предоставляет возможность двигать объектами прямо на определенной скорости.

3) Scroll to или Прокручивание, необходим для прокручивания экрана и следует за объектом, когда он перемещается (также известен как scrolling).

4) Bound to layout или Ограничение по плану данная функция не позволяет объекту выходить за границы плана.

5) Destroy outside layout или Уничтожение объектов за границами плана, необходим для того чтобы останавливать объект, когда он уходит за границы плана. Такая функция уничтожает объект и удобен для пули. При отсутствии данной функции пули улетали бы с экрана и поглощали бы больше памяти и мощности процессора.

6) Fade или Затухание, такая функция создает эффект постепенного исчезания, используется для взрывов.

Плагины определяют вид объекта. Например, Sprite - будет объектом одного вида, а Audioobject - будет уже объектом другого вида. Они определяются плагином Sprite и Audioobject соответственно. Программисты Javascript могут создавать новые плагины (и модели поведения) с помощью JavaScript SDK.

Существует три основных вида плагинов:

1) Визуальные плагины (Sprite) появляются в макете и нарисовать что-то на экране.

2) Скрытые плагины (Array) размещаются в определенном макете, но ничего не рисуют на экране.

3) Плагины для всего проекта (Mouse, Audio) добавляются ко всему проекту и могут быть добавлены только один раз.

Программа спроектирована модульно. Это означает, что встроено не так уж много функциональности: можно вставить плагин, прежде чем появится возможность использовать связанные с ним функции. Например, нельзя воспроизвести какой-либо звук до добавления звукового плагина в проект. Существует также множество плагинов для всего проекта, но не каждый проект будет использовать их все. Например, если аудио-плагин был автоматически включен в каждый проект, даже в тот, который не будет нуждаться в Аудио поддержке это повлияет на его производительность и будет являться причиной большой длительностью по загрузке и прочее. Поэтому если тот или иной плагин не включается в проект - это помогает добиться большей производительности.

Объект Sprite - это анимируемое изображение, которое появляется в игре. Это один из самых важных объектов для большинства игр Construct 2. Они используется для создания большинства визуальных элементов в игре, таких как игрок, враги, снаряды, взрывы и декорации.

Свойства спрайта:

1) Анимация

2) Размер

3) Начальная видимость

4) Начальная анимация

5) Начальный кадр

6) Столкновения

Event Sheet - это списки событий. Они редактируются в представлении Event Sheet View. Все таблицы событий в проекте перечислены в Панели проектов. Листы событий можно переименовать или удалить, а также возможно совместное использование событий между макетами. Макеты имеют связанный лист событий для определения того, как работает макет. Однако часто бывает полезно использовать один лист событий для нескольких макетов, чтобы избежать необходимости дублировать все события.

Разработка приложений под Android - одно из очень популярных направлений в программировании. И рынок только продолжает расти поэтому языков программирования, востребованных среди Android - разработчиков довольно много.

Java лидирует в этой сфере. Его можно смело назвать основным официальным языком разработки Android. Потому что большинство официальных курсов и образовательная документация по Android-программированию основана на этом языке. На Java существует огромное количество исходников на GitHub, да и сами разработчики отмечают, что этот красивый и мощный язык очень удобен для написания мобильных приложений. Так как языку уже 22 года, конечно, это не самый быстрый и простой процесс, а простота никогда не была его преимуществом. Существует мнение, что при желании можно обойтись более современными аналогами. Однако практика показывает, что без Java добиться успехов в сфере создания приложений под Android еще никому не удалось [1.6].

Вопреки распространенному и вполне обоснованному скептическому отношению к решениям Microsoft, язык C# заслуживает пристального

внимания. Эта разработка вобрала в себя все лучшее, что есть в Java. Одновременно были устранены многие недостатки. Для работы под Android были созданы функциональные и очень удобные среды программирования Visual и Xamarin Studio. Также C# вам пригодится и станет большим плюсом, когда вы начнете использовать Unity3D. Такой набор расширяет ваши возможности практически до безграничных.

Язык Python считается неподходящим для создания нативных Android приложений, так как его не поддерживает. Однако нет ничего невозможного и ценители языка Python создали много инструментов, которые помогут скомпилировать программы Python в нужный код. Одним из наиболее востребованных фреймворков считается Kivy, в этой среде можно создать полноценное приложение для PlayMarket полностью на языке Python. В профессиональных сообществах Python всегда найдутся люди, которые помогут овладеть этим инструментом и подскажут нестандартные решения [1.6].

Kotlin является одним из молодых языков, уже заслуживших популярность. В действительности он достоин пристального внимания и сам по себе, и в связке с Java. Также несмотря на юный возраст программисты не сумели найти в Kotlin практически никаких недостатков, а в его родной среде IntelliJ IDEA вы сможете быстро и комфортно создавать нативные полноценные Android-приложения. Однако тех, кто знаком с Kotlin очень мало, да и спрос на них только формируется. Однако язык очень перспективный, и его изучение станет прекрасным «вложением» в будущее.

Разработчиками мобильных приложений также очень рекомендуется освоить языки веб-программирования, как минимум, стандартный базовый набор, состоящий из HTML, CSS и JavaScript. Без них возможности сильно ограничены разработкой узкоспециализированных приложений, а в большинстве случаев нужны гибридные решения, в том числе, под Web. Для работы с веб-версиями можно применять среду PhoneGap Build или, в отдельных случаях, Adobe Cordova. Глубоких знаний в этой сфере от

разработчика мобильных приложений обычно не требуется, однако определенная база - очень важна [1.4].

Lua язык программирования, который старше популярного Java. Считается, что Lua постепенно уходит в прошлое, но тем не менее, сегодня в среде разработки под Android, он все еще востребован. Среди его преимуществ специалисты подчеркивают динамическую типизацию, а также простой и понятный синтаксис. Применяют большей частью в игровых приложениях. Он удобен для создания прослойки программного кода между оболочкой и движком, а потому он традиционно применяется для адаптации игровых проектов для мобильных устройств. Наиболее популярная среда разработки мобильных приложений при помощи Lua - это Corona SDK. Распространяется она полностью бесплатно (с 2015 года) и считается очень удобной для начинающих разработчиков. По ней много документации и полезных советов от специалистов, в том числе и в рунете [1.6].

1.3. Формализованное описание технического задания

на разработку игрового приложения в жанре 2D-платформера под операционную систему Android

Составлен на основе ГОСТ 34.602-89 «Техническое задание на создание автоматизированной системы».

1. Общие сведения.

1.1. Название организации-заказчика.

ФГБОУ ВО «УрГПУ»

1.2. Название продукта разработки (проектирования).

Игровое приложение в жанре 2D-платформера под операционную систему Android.

1.3. Назначение продукта.

Игровое мобильное приложение, обучающее мобильное приложение для сайта посвященного компьютерной графике.

1.4. Плановые сроки начала и окончания работ.

В соответствии с планом выполнения ВКР (01.12.2020 - 01.03.2021).

2. Характеристика области применения продукта.

2.1. Процессы и структуры, в которых предполагается использование продукта разработки.

Игровое мобильное приложение, обучающее мобильное приложение предназначено для размещения на веб-странице в сети интернет.

2.2. Характеристика персонала (количество, квалификация, степень готовности)

Разработчик (Должен владеть навыками работы с ПК, навыками владения программной среды разработки - Construct 2, Cordova, Android Studio)

Пользователь (владеть навыками работы с векторными графическими редакторами).

3. Требования к продукту разработки.

3.1. Требования к продукту в целом.

Предоставить мобильное игровое приложение в жанре 2D-платформера, на основе которого будет реализовано учебное пособие в виде мобильного обучающего приложения. Мобильное игровое приложение должно содержать в себе: основное меню с функцией старта и выхода, меню выбора уровня, один игровой персонаж, платформы для передвижения, платформы для прыжка, движущиеся платформы, система бонусов, противник с функцией «патрулирования», кнопки управления. Мобильное обучающее приложение должно содержать в себе: выдвижное меню выбора, блок выбора «урока» содержащий визуальную (картинки) и текстовую информацию. Размещение продукта на веб-странице размещенном на веб-сервере.

3.2. Аппаратные требования.

Мобильное устройство с поддержкой платформ: ARM, MIPS, x86, x86_64. Оперативная память не менее 256 мб. Процессор с тактовой частотой не менее 768 МГц.

3.3. Указание системного программного обеспечения (операционные системы, браузеры, программные платформы и т.п.).

Операционная система Android версии от 6.0.1 (Marshmallow).

3.4. Указание программного обеспечения, используемого для реализации.

Программная среда разработки: Construct 2, Unity 3D, Unreal Engine 4, Game Maker.

4. Требования к пользовательскому интерфейсу.

4.1. Общая характеристика пользовательского интерфейса.

Общая концепция игрового приложения «Персонаж Arduino собира-ет светодиоды, избегая врага (ошибку)». Индивидуально разработанный дизайн персонажа, врага, платформ, бонуса.

4.2. Размещение информации на экране, дизайн экрана.

См. Приложение 1

5. Требования к документированию.

5.1. Перечень сопроводительной документации.

Не предусмотрено.

6. Порядок сдачи-приемки продукта.

В соответствии с планом выполнения ВКР.

1.4. Назначение продукта разработки

Продуктом разработки данной выпускной квалификационной работы является игровое приложение в жанре 2D-платформера. Приложение разрабатывается в программе Construct 2, что является одной из наиболее простых сред разработки под операционную систему Android, с которой может справиться каждый, а это значит, что каждому человеку доступна возможность разработать игру для получения выгоды.

Программа легка в освоении, а знания графических редакторов поможет создать индивидуальный дизайн, присущий только конкретному автору приложения, но даже если человек не силен в графических редакторах - это не повод отказаться от разработки, ведь данные материалы можно найти в свободном доступе или же приобрести за определенную плату. Освоив такую программу человек может задуматься о более широких возможностях и

освоить уже не только данную программу, но и другие языки разработки и игровые движки, чтобы создать более основательный продукт.

Так как продуктом разработки является игровое приложение в жанре 2D-платформера, а программой в которой оно было разработано Construct 2 - то было принято решение оформить данный продукт в методическое пособие в виде обучающего Android-приложения, которые будут созданы в процессе выполнения данной выпускной квалификационной работы и вместе с игровым приложением будут представлять из себя готовый продукт.

Так как знания графических редакторов позволяют создать индивидуальный дизайн, мобильное игровое приложение и разработанное на основе игры – обучающее приложение будут расположены на веб-сайте посвященному Компьютерной графике, как основа для применения полученных знаний в реализации продукта.

ГЛАВА 2. РАЗРАБОТКА ИГРОВОГО ПРИЛОЖЕНИЯ

2.1. Описание процесса разработки игрового приложения

Открыв новый проект, создание игры начинается с добавления такого объекта как «sprite». Он используется для создания большинства визуальных элементов в игре, таких как игрок, враги, снаряды, взрывы и не-плиточные декорации (плиточные декорации гораздо лучше делаются с плиточным фоновым объектом).

Если у «sprite» есть одна анимация с одним кадром, он просто показывает изображение без анимации. Однако с помощью редактора изображений и анимаций к объектам «sprite» можно добавить несколько анимаций.

Все экземпляры объектов «sprite» имеют общую анимацию. Другими словами, существует один набор изображений, содержащий анимацию, которая принадлежит объектному типу, и на эти изображения ссылаются экземпляры.

В первую очередь следует разместить платформы, по которым будет передвигаться персонаж и враг. Всего таких платформ будет несколько - обычные, движущаяся и для прыжка.

Расставив все платформы в необходимом порядке, можно перейти к созданию игрока, добавив его с помощью «sprite». Игроку нужно задать поведение, то есть «behaviours» платформера, благодаря которому персонаж может двигаться.

Чтобы игрок соприкасался с платформой и не падал сквозь неё, платформе следует задать твердость с помощью поведения «solid».

Чтобы игрок был не бесформенным квадратом, ему следует задать оболочку, у которой будет анимация движения - ходьбы, прыжка и падения, а также просто положение неактивности.

Графическая основа готова, осталось задать активности персонажу и платформам, благодаря которым будет осуществляться механика игрового приложения. В списке событий «event sheet» добавляются события «add event».

Первое событие, которое необходимо добавить «every tick» для того чтобы привязать оболочку к персонажу с помощью «set position to another object».

Последующие события будут отвечать за движение персонажа, где им будет задано определенное поведение и анимация данного поведения из оболочки персонажа.

Исходя из всего этого подобным образом будут добавлены все движения персонажа и анимация к ним, а также ограничения для персонажа в виде того, что, когда он не прыгает и не падает - будет применяться анимация ходьбы, а если не двигается будет приниматься анимация неактивного состояния.

Добавив «sprite» - клавиатура, благодаря которому персонажу передвигается, необходимо задать событие положения передвижения влево и вправо, чтобы он мог двигаться не исключительно в одну сторону в связи со

своей анимацией, но и отраженно если будет задано другое положение при ходьбе.

Добавляем одной из платформ событие прыжка, чтобы при соприкосновении с данной платформой персонаж прыгал по ней без осуществления действий со стороны игрока.

Добавляем системное событие, благодаря которому камера будет двигаться вслед за персонажем «scroll to y», то есть скролинг по оси y.

В большинстве игровых платформеров добавляется игровой объект, который игрок будет собирать, так называемая монетка или в случае с данной игрой монеткой будет служить светодиод.

К данному игровому объекту должно применяться действие «on collsion with another object», которое будет отвечать за уничтожение монетки после соприкосновения с игроком.

Чтобы велся счет собранных светодиодов, необходимо добавить текстовый объект.

Данному текстовому объекту следует задать переменную «instance variables», которая будет отвечать за отражение количества очков.

Событие, которое будет отвечать за подсчет очков является системным и запускается вместе с текущим уровнем «on start of layout».

Этому событию присваивается активность «score», а ей присваивается переменная «score», таким образом будет производиться изменение числа в большую сторону. Такую же активность присваиваем игроку при соприкосновении со светодиодом, при уничтожении которого будет прибавляться 1 очко.

Чтобы светодиод не просто располагался на карте, а двигался - ему необходимо добавить поведение «sine», после задать координату движения - по вертикали и магнитуду, подобным же образом настраивается движение платформы - по горизонтали с магнитудой 120.

Следующий шаг - добавление врага, он будет патрулирующим, что означает - будет взаимодействовать с персонажем и вычитать ему жизни, но

игрок не сможет его сразить. Добавление врага ничем не отличается от добавления игрока, но врагу нужно выставить ограничение, чтобы он передвигался в строго установленных рамках. Данным границам задается параметр невидимости, чтобы в самой игре их было не видно. Такой точке задается переменная «rotate», чтобы враг при приближении к границе разворачивался и двигался в другую сторону - влево или вправо. Также необходимо добавить небольшой промежуток ожидания, чтобы враг двигался-ждал-поворачивался-двигался.

Теперь после добавления врага, необходимо добавить игроку активность, при которой соприкасаясь с врагом - игрок бы терял жизнь.

После того как было настроено соприкосновение с врагом, необходимо настроить перезапуск уровня, когда количество жизней достигнет значение нуля, при котором игрок будет уничтожен на уровне и уровень начнется сначала.

С помощью «tiled background» добавляем текстуру сердца и задаем ей параметр 30 на 30, в настройках «common» задаем параметры высоты и ширины 90 на 30 - таким образом на карте будет отображаться 3 сердца.

Далее добавляем игроку активность - при соприкосновении с объектом сердце количество жизней будет увеличиваться на одну позицию, что в «tiled background» будет означать прибавление на 30 в ширину и появление нового сердца в количестве жизней. Сердце будет соответственно уничтожаться при соприкосновении с игроком.

Добавляем игроку поведение «flash» тогда игрок будет мигать при соприкосновении с игроком и потере жизни.

Теперь добавим ряд «sprite», первый будет отвечать за фон и второй, который будет отвечать за переход к новому уровню.

Далее добавляем игроку активность - при соприкосновении с объектом, отвечающим за переход на новый уровень, будет вызываться системное событие, отвечающее за переход к новому уровню.

Для того чтобы отвечать за движение персонажа на платформе android, необходимо добавить кнопки управления, а к ним объект «touch». Добавив кнопки управления с помощью «sprite» - им задается активность симуляции движения влево, вправо и прыжок.

2.2. Описание структуры обучающего приложения

Приложения для android могут быть простыми и сложными, но строение приложений всегда будет одинаковым, в них есть обязательные элементы приложений, а есть опциональные, которые используются по мере необходимости. Само приложение состоит из нескольких основных компонентов: манифест приложения, набор различных ресурсов и исходный код программы.

Обязательные и возможные составляющие структуры Android-приложения: Gen -файлы, сгенерированные самой Java. Здесь находится такой важный файл как R.java; AndroidManifest.xml - файл манифеста AndroidManifest.xml предоставляет системе основную информацию о программе. Каждое приложение должно иметь свой файл манифеста; Src - каталог, в котором содержится исходный код приложения; Assets - произвольное собрание каталогов и файлов; Res - каталог, содержащий ресурсы приложения. В данном каталоге могут находиться подпапки drawable, anim, layout, menu, values, xml.

Приложение состоит из выдвижного меню, экрана с выбором, экрана с текстовой информацией

Сначала создаем новый проект и используем для основы Navigation Drawer Activity.

Далее именуем проект, выбираем язык программирования Java и версию Android от которой возможна установка на устройство.

В открывшемся интерфейсе и начинается вся основная работа по написанию приложения. Где основой является манифест, без него приложение не может осуществлять работу.

Манифест или AndroidManifest.xml предоставляет системе основную информацию о программе, каждое приложение должно иметь свой файл AndroidManifest.xml. Редактировать файл манифеста можно вручную, изменяя XML-код или через визуальный редактор Manifest Editor, который позволяет осуществлять визуальное и текстовое редактирование файла манифеста приложения.

Назначение файла: описывает компоненты приложения - Activities, Services, Broadcast receivers и Content providers; содержит список необходимых разрешений для обращения к защищенным частям API и взаимодействия с другими приложениями; объявляет разрешения, которые сторонние приложения обязаны иметь для взаимодействия с компонентами данного приложения; объявляет минимальный уровень API Android, необходимый для работы приложения; перечисляет связанные библиотеки.

В Android-приложениях, пользовательский интерфейс построен на View и ViewGroup объектах. Класс ViewGroup является основой для подкласса Layout (разметка).

Компоновка (также используются термины макет или разметка) хранится в виде XML-файла в папке /res/layout. Так сделано для того, чтобы отделить код от дизайна, как это принято во многих технологиях (HTML, CSS и другие). Помимо основной компоновки для всего экрана, существуют дочерние компоновки для группы элементов, что по сути означает некий визуальный шаблон для пользовательского интерфейса приложения, который позволяет управлять элементами, их свойствами и расположением.

Приложение состоит из таких layout как activity_main.xml, который содержит выдвижное меню.

Layout app_bar_main.xml содержит элемент toolbar, который позволяет пользователю получить доступ к используемым функциям.

Layout content_main.xml и main.xml содержат ListView - это view group, отображающий элементы в виде списка, который можно прокручивать вертикально.

Layout text_content.xml содержит: ScrollView, который служит для упаковки иерархии объектов View в контейнер его можно прокручивать (скролить); LinearLayout, который выравнивает все дочерние объекты в одном направлении — вертикально или горизонтально; компонент ImageView, который предназначен для отображения изображений; компонент TextView предназначен для отображения текста без возможности редактирования его пользователем.

Меню в приложении можно задать как XML-ресурсы и в приложении меню является activity_main_drawer.xml, на который ссылается activity_main.xml, так как информация из activity_main_drawer.xml отображается в выдвижном меню.

Текст, который видит пользователь прописывается в strings - строковом ресурсе, а изображения, отображаемые в приложении, содержатся в drawable - ресурсе-изображения, оно поддерживает форматы JPG, GIF, PNG (самый предпочтительный) и другие. Все изображения являются отдельными файлами. Система также поддерживает stretchable images, в которых можно менять масштаб отдельных элементов, а другие элементы оставлять без изменений.

Activity представляет собой пользовательский интерфейс для одного действия, которое пользователь может совершить. Например, приложение для обмена текстовыми сообщениями может иметь одно Activity для отображения списка контактов, другое - для написания сообщения выбранному контакту, третье - для просмотра сообщений и еще одно для изменения настроек. Все эти Activities формируют единый пользовательский интерфейс, но не зависят друг от друга.

Жизненный цикл activity начинается с вызова метода onCreate(), в котором производится первоначальная настройка глобального состояния, и завершается вызовом метода onDestroy(), в котором оно освобождает занятые ресурсы, его видимая часть происходит между вызовами onStart() и onStop(). В течение этого времени пользователь может видеть Activity на экране, хотя

оно может быть не на переднем плане и не взаимодействовать с пользователем, а сами методы onStart() и onStop() могут вызываться столько раз, сколько Activity становится видимым или скрытым.

На переднем плане Activity находится между вызовами onResume() и onPause(), в течение этого времени оно находится поверх других и взаимодействует с пользователем. Сам Activity может часто переходить в состояние паузы и выходить из него. К примеру, метод onPause() может быть вызван, когда устройство переходит в спящий режим или когда запускается другое Activity, а метод onResume() - при получении результата от закрывающегося Activity.

Приложение состоит из Activity MainActivity.java, который содержит весь жизненный цикл приложения и Activity Text_Content_Activity.java, который содержит жизненный цикл информации, которую увидит пользователь.

ЗАКЛЮЧЕНИЕ

В процессе выполнения работы анализ операционная система Android, показал её архитектурные особенности, пользовательский интерфейс, среду программирования для разработки игр. Ключевым моментом работы стал разработанный продукт - игровое приложение для использования его в качестве методического пособия.

Для того чтобы разработанное игровое приложение можно было использовать в качестве методического пособия, была рассмотрена программа Construct 2, позволяющая разрабатывать игры с минимальным знанием программного кода, и которая является конструктором, который на основе игрового движка и интегрированной среды разработки, представляет собой представляет собой визуальный редактор, в котором имеются события и действия, создающие логику игры.

Предметом разработки являлся процесс создания игрового приложения, на основе которого было создано методическое пособие в виде приложения для операционной системы Android. Для разработки этого исследовательского процесса во время работы было создано также персонализированное графическое оформление, которое является индивидуальной разработкой.

Целью данной выпускной квалификационной работы являлась разработка продукта - данная цель была выполнена, и создание методического пособия - данная цель была осуществлена. Все поставленные задачи были выполнены в полной мере и результат работы доступен на сайте посвященном компьютерной графике (см. Приложение 4).

СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

1. Литература

1.1. Хашими С., Коматинени С., Маклин Д. Разработка приложений для Android. - СПб.: Питер, 2011.

1.2. Голощапов А. Л. Google Android: программирование для мобильных устройств. - 2-е изд., перераб. и доп. - СПб.: БХВ-Петербург, 2012.

1.3. Андерс Ёранссон. Эффективное использование потоков в операционной системе Android. Технологии асинхронной обработки данных. - СПб.: ООО «ДМК», 2017.

1.4. Гриффитс Дэвид Марк, Гриффитс Дон. Head First. Программирование для Android. 2-е издание. — СПб.: Питер, 2018.

1.5. Дейтел Пол, Дейтел Харви, Уолд Александер. Android для разработчиков. 3-е издание. - СПб.: Питер, 2016.

1.6. И.Н. Блинов, В.С. Романчик. Java. Методы программирования: учебно-методическое. — Минск: издательство «Четыре четверти», 2016.

1.7. Шилдт Герберт. Java. Руководство для начинающих. Современные методы создания, компиляции и выполнения программ на Java. - СПб.: Диалектика, 2018.

1.10. Рейтц К., Шлюссер Т. Автостопом по Python . — СПб.: Питер, 2017.

1.11. Фултон Х., Арко А. Путь Ruby. - СПб.: Питер, 2015.

1.12. Гриффитс Дон, Гриффитс Дэвид. Head First. Программирование для Android. — СПб.: Питер, 2016.

1.13. Медникс З., Дорнин Л., Мик Б., Накамура М. Программирование под Android. 2-е изд. — СПб.: Питер, 2013

1.14. ГОСТ 34.602-89. Информационная технология. Комплекс стандартов на автоматизированные системы. Техническое задание на создание автоматизированной системы. - М.: Стандартинформ, 2009.

2. Интернет-ресурсы

2.1. Интуит. - URL: https://www.intuit.ru/studies/courses/

4462/988/lecture/14988?page=1 (дата обращения: 04.02.2021).

2.2. Trashbox. Высокие технологии. - URL: https://trashbox.ru/topics/109325/istoriya-razvitiya-mobilnyh-igr (дата обращения: 04.02.2021).

2.3. Java программирование и разработка под Android. - URL: https://javadevblog.com/primer-ispol-zovaniya-scrollview-v-android.html (дата обращения: 04.02.2021).

2.4. Devcolibri. Android для начинающих. - URL: https://devcolibri.com/unit/урок-11-работа-с-toolbar-и-menu-на-примере-userinfoactivity (дата обращения: 04.02.2021).

2.5. Programming tutorial: Java, Android, C/C++, C#, SQL. - URL: https://o7planning.org/ru/10435/android-listview-tutorial (дата обращения: 04.02.2021).

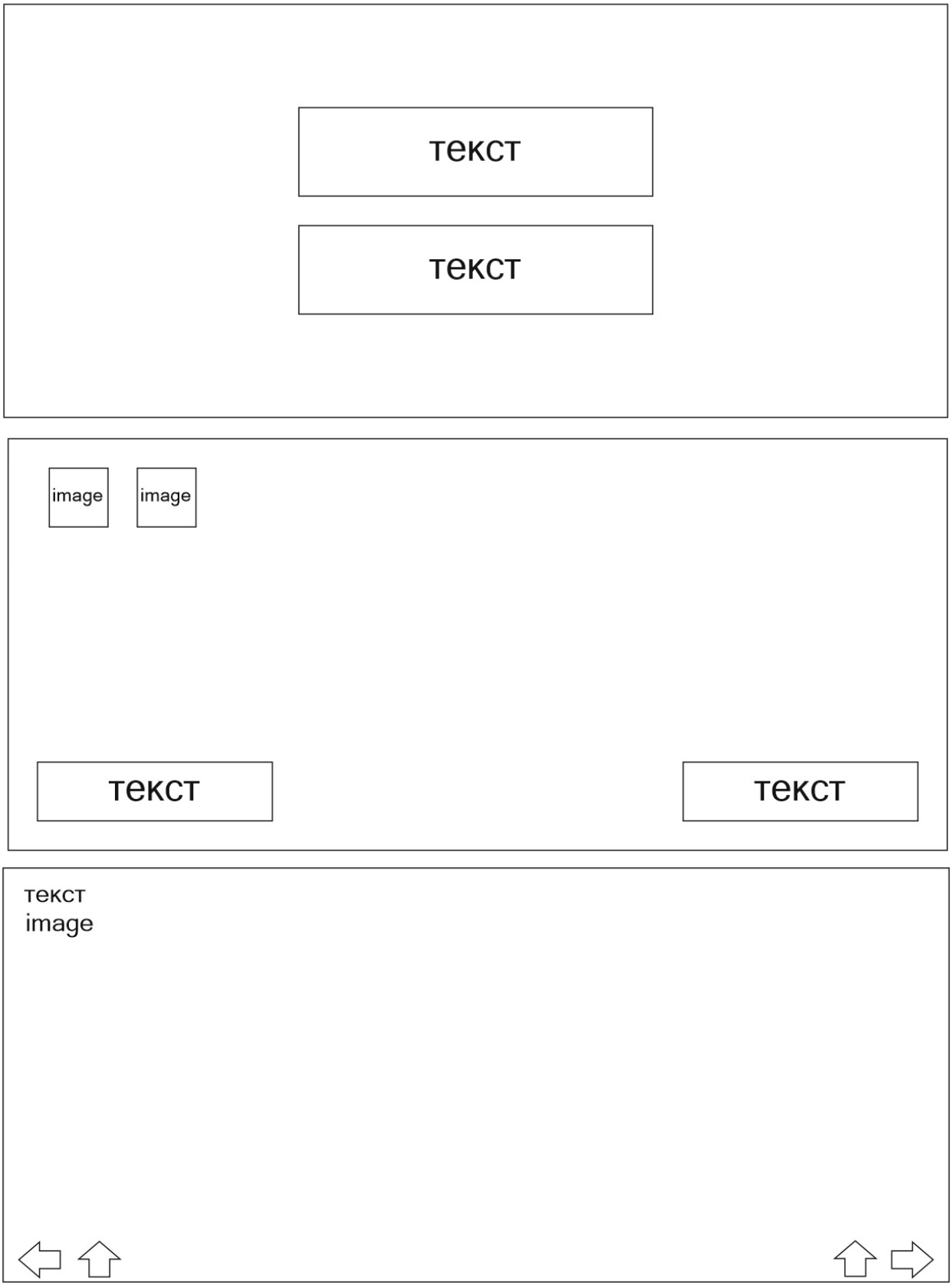
2.6. Mob-mobile. - URL: https://mob--mobile-ru.turbopages.org/mob-mobile.ru/s/programs/1639-programma-dlya-sozdaniya-igr-na-telefon-obzor.html (дата обращения: 04.02.2021).

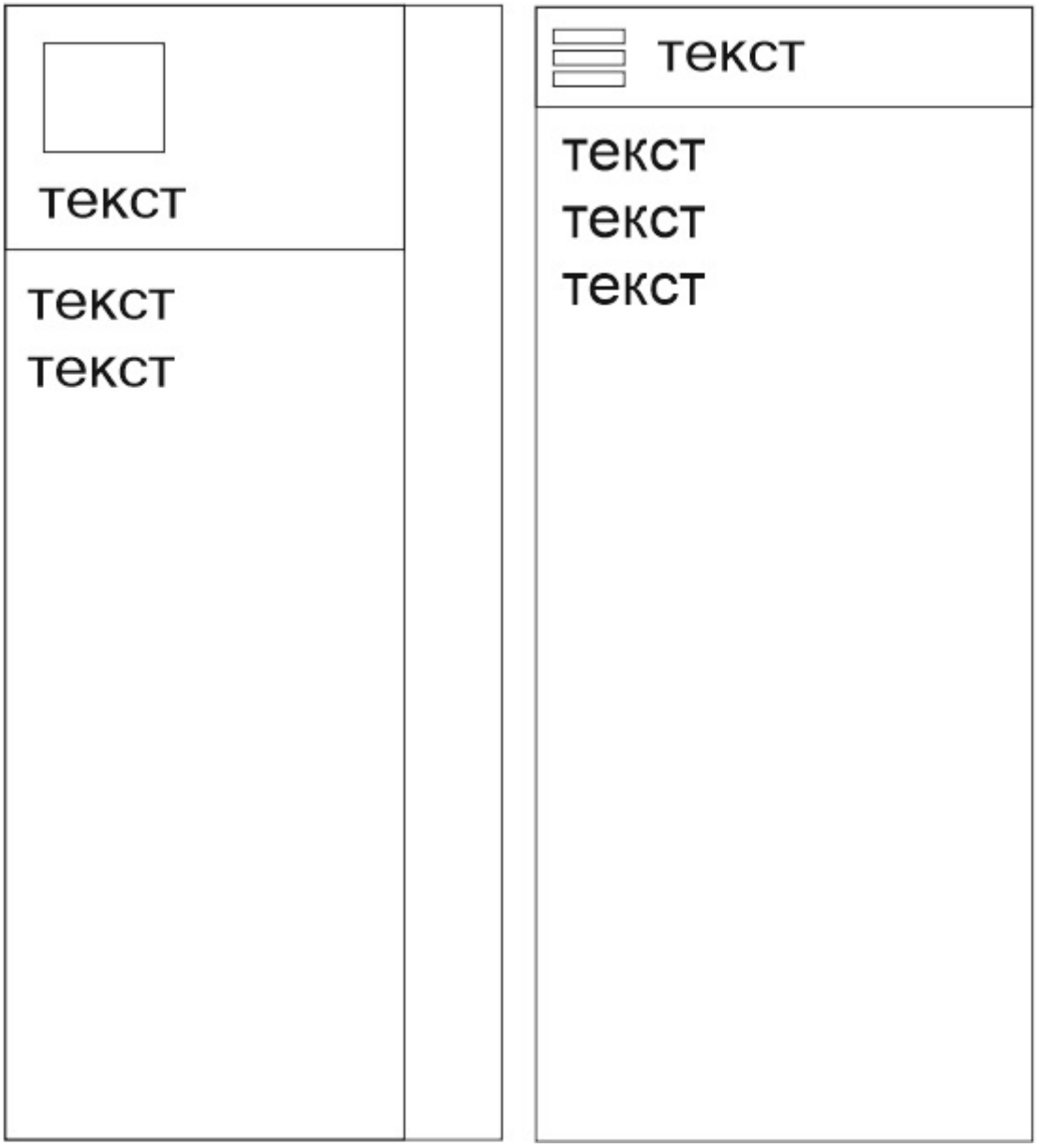
2.7. Хабр. - URL: https://habr.com/ru/post/250703/ (дата обращения: 04.02.2021).

2.8. Студопедия. - URL: https://studopedia.ru/25_69828_paketi-kotorie-yavlyayutsya-chastyu-Android-SDK.html (дата обращения: 04.02.2021).

2.9. Академик. - URL: https://dic.academic.ru/dic.nsf/ruwiki/1042635 (дата обращения: 04.02.2021).

ПРИЛОЖЕНИЕ 1




текст
текст
текст
текст
текст
текст

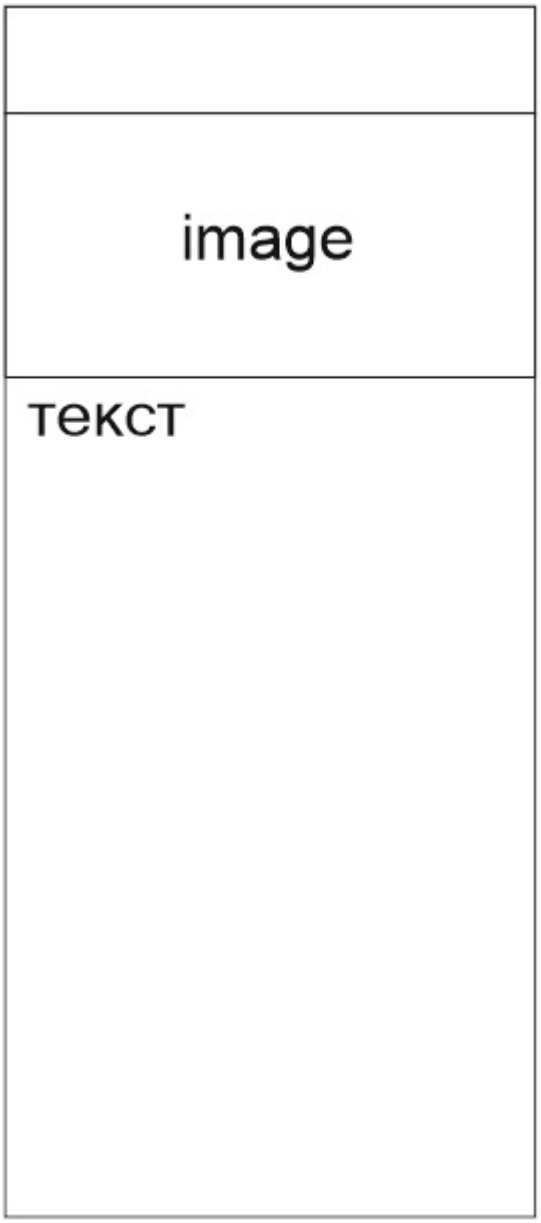
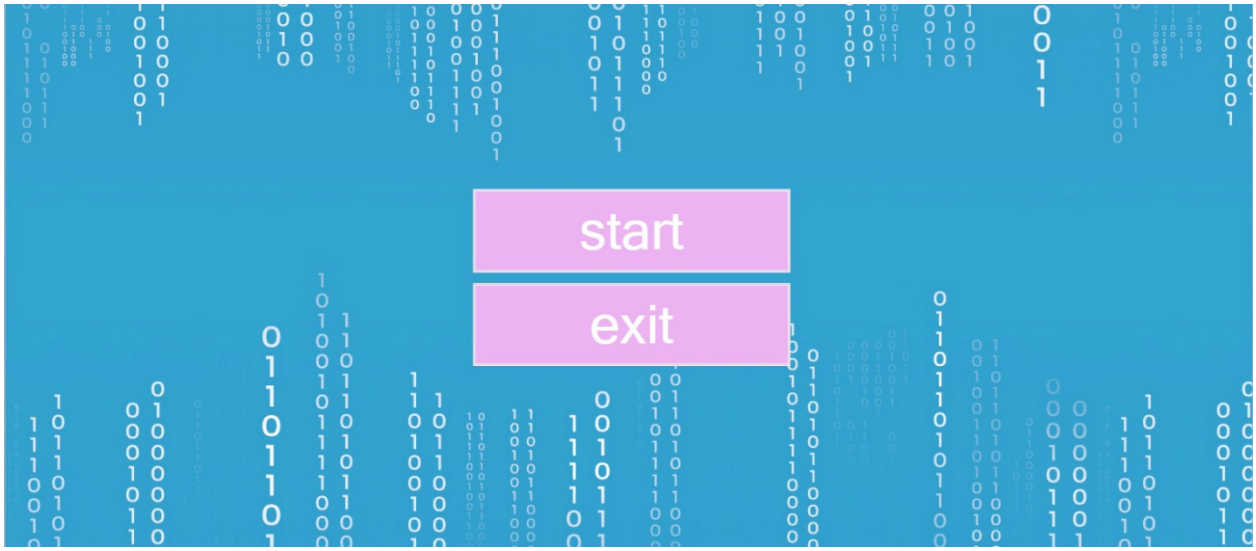
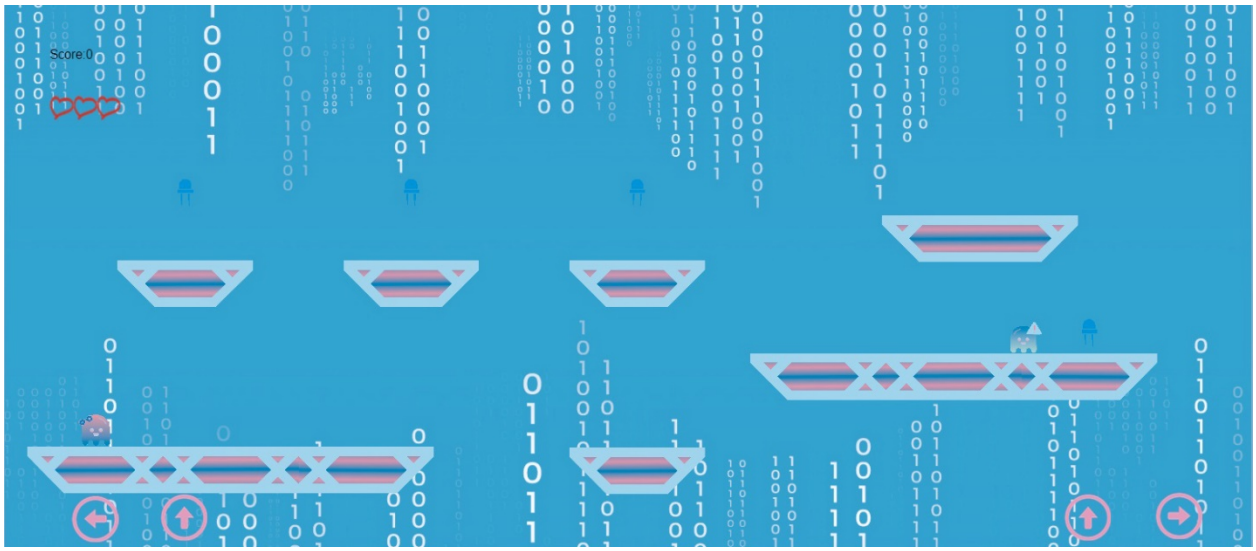


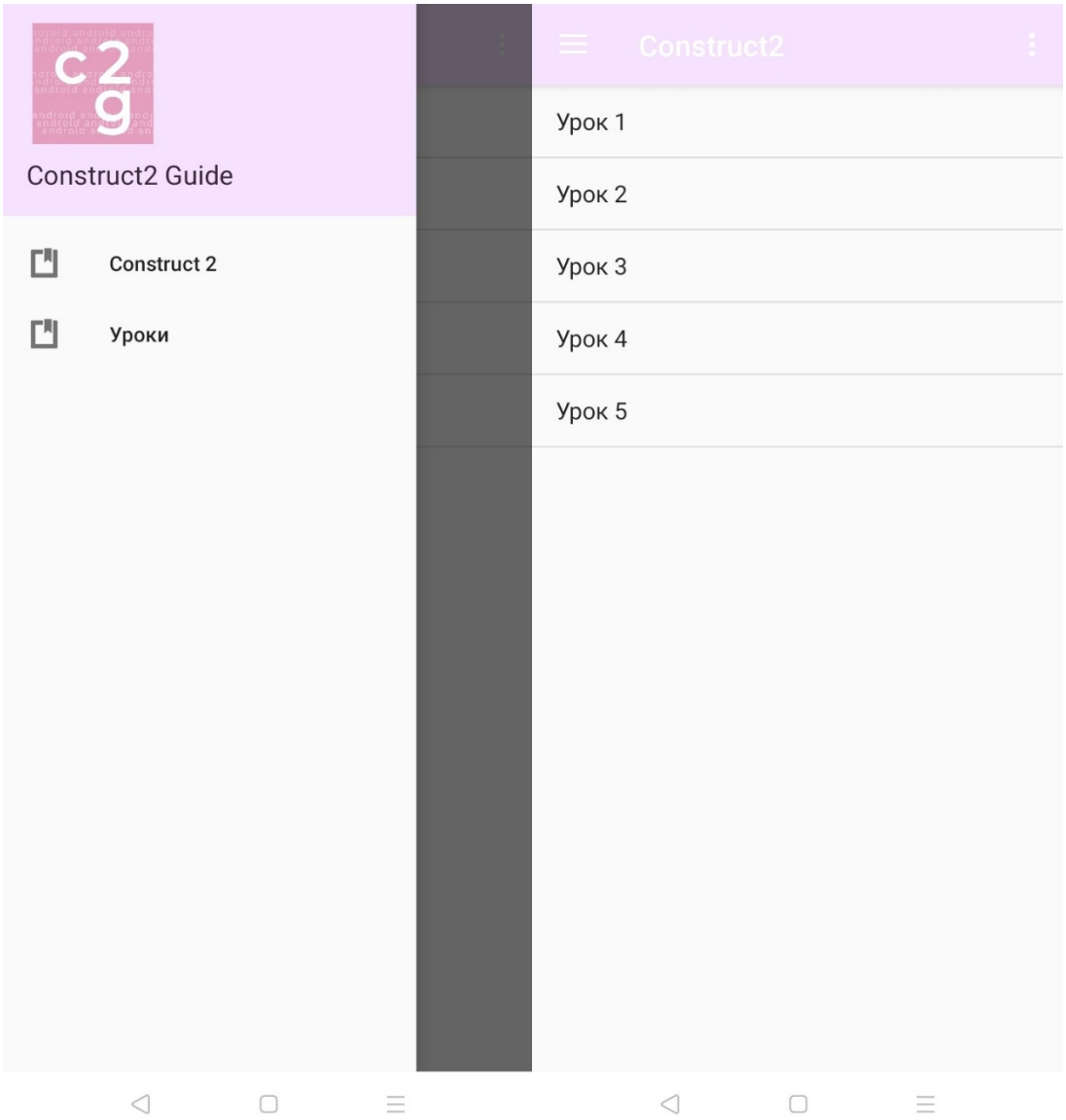
image
текст

ПРИЛОЖЕНИЕ 2

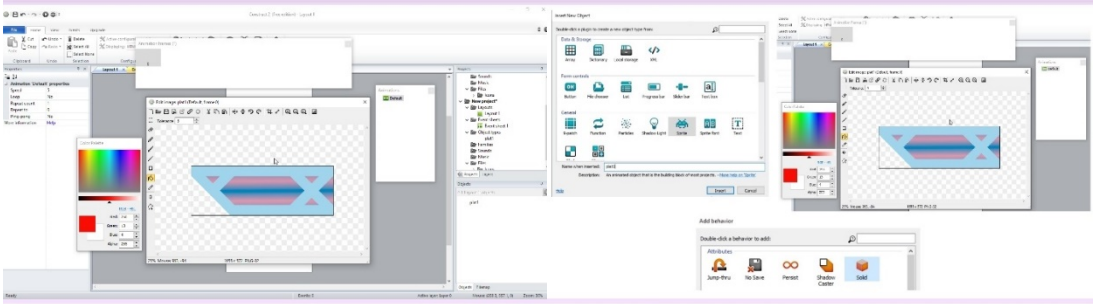




ПРИЛОЖЕНИЕ 3



Graphic guide



Поведение платформы реализует движение в стиле бокового обзора прыжок и бег. Он поддерживает наклоны, движущиеся платформы, прыгающие платформы и произвольные углы тяжести. Существует несколько примеров поведения платформы, которые можно найти в диалоговом окне пуск. Поведение платформы будет приземляться на любые объекты с твердым или прыжковым поведением. Jump-through отличается тем, что движение платформы может прыгать на Jump-through снизу, в то время как прыжок на твердое тело снизу заставляет игрока отскакивать. Свойства платформы:Максимальная скорость.Максимальная скорость пола в пикселях в секунду;Ускорение. Ускорение горизонтального движения в пикселях в секунду в секунду; Замедление. Замедление горизонтального движения в пикселях в секунду в секунду. При движении в направлении, противоположном направлению движения, ускорение и замедление объединяются; Сила прыжка. Начальная вертикальная скорость прыжка в пикселях в секунду при нажатии клавиши прыжка; Гравитация. Ускорение, вызванное гравитацией, выражается в пикселях в секунду в секунду; Максимальная скорость падения. Максимальная скорость в пикселях в секунду, до которой объект может разогнаться в свободном падении; Двойной прыжок. Если эта

Открыв новый проект, создание игры начинается с добавления такого объекта как sprite. Объект sprite - это анимируемое изображение, которое появляется в игре. Это один из самых важных объектов для большинства игр Construct 2. Он используется для создания большинства визуальных элементов в игре, таких как игрок, враги, снаряды, взрывы и не-плиточные декорации (плиточные декорации гораздо лучше делаются с плиточным фоновым объектом). Если у sprite есть одна анимация с одним кадром, он просто показывает изображение без анимации. Однако с помощью редактора изображений и анимаций к объектам sprite можно добавить несколько анимаций. Все экземпляры объектов sprite имеют общую анимацию. Другими словами, существует один набор изображений, содержащий анимацию, которая принадлежит объектному типу, и на эти изображения ссылаются экземпляры. В первую очередь следует разместить платформы, по которым будет передвигаться персонаж и враг. Всего таких платформ будет несколько – обычные, движущаяся и для прыжка. Чтобы игрок соприкасался с платформой и не падал сквозь неё, платформе следует задать твердость с помощью поведения «solid».

ПРИЛОЖЕНИЕ 4